Tutorial 8 - Options

Please complete this tutorial to get an overview of options and an implementation of SMDP Q-Learning and Intra-Option Q-Learning.

References:

Recent Advances in Hierarchical Reinforcement Learning is a strong recommendation for topics in HRL that was covered in class. Watch Prof. Ravi's lectures on moodle or nptel for further understanding the core concepts. Contact the TAs for further resources if needed.

```
!pip install numpy==1.23

Requirement already satisfied: numpy==1.23 in
/usr/local/lib/python3.10/dist-packages (1.23.0)

!pip install gym==0.22

Collecting gym==0.22
  Downloading gym-0.22.0.tar.gz (631 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 631.1/631.1 kB 3.7 MB/s eta
0:00:00
ents to build wheel ... etadata (pyproject.toml) ... ent already
satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages
(from gym==0.22) (1.23.0)
Requirement already satisfied: cloudpickle>=1.2.0 in
/usr/local/lib/python3.10/dist-packages (from gym==0.22) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in
/usr/local/lib/python3.10/dist-packages (from gym==0.22) (0.0.8)
Building wheels for collected packages: gym
  Building wheel for gym (pyproject.toml) ... : filename=gym-0.22.0-
py3-none-any.whl size=708362
sha256=6639a8a65776e73e52aa3b1e41b5d6fd26e9447de6f482e35aff13fbce3e440
d
  Stored in directory:
/root/.cache/pip/wheels/42/e8/e8/6dfbc92a1dcd76c1a5e2bb982750fd6b7e792
239f46039e6b1
Successfully built gym
Installing collected packages: gym
  Attempting uninstall: gym
    Found existing installation: gym 0.25.2
    Uninstalling gym-0.25.2:
      Successfully uninstalled gym-0.25.2
Successfully installed gym-0.22.0

'''
A bunch of imports, you don't have to worry about these
'''
```

```
import numpy as np
import random
import gym
from gym.wrappers import Monitor
import glob
import io
import matplotlib.pyplot as plt
from IPython.display import HTML


'''
The environment used here is extremely similar to the openai gym ones.
At first glance it might look slightly different.
The usual commands we use for our experiments are added to this cell
to aid you
work using this environment.
'''

#Setting up the environment
from gym.envs.toy_text.cliffwalking import CliffWalkingEnv
env = CliffWalkingEnv()

env.reset()

#Current State
print(env.s)

# 4x12 grid = 48 states
print ("Number of states:", env.nS)

# Primitive Actions
action = ["up", "right", "down", "left"]
#correspond to [0,1,2,3] that's actually passed to the environment

# either go left, up, down or right
print ("Number of actions that an agent can take:", env.nA)

# Example Transitions
rnd_action = random.randint(0, 3)
print ("Action taken:", action[rnd_action])
next_state, reward, is_terminal, t_prob = env.step(rnd_action)
print ("Transition probability:", t_prob)
print ("Next state:", next_state)
print ("Reward recieved:", reward)
print ("Terminal state:", is_terminal)
env.render()
```

```
36
Number of states: 48
```

```
Number of actions that an agent can take: 4
Action taken: left
Transition probability: {'prob': 1.0}
Next state: 36
Reward recieved: -1
Terminal state: False
o  o  o  o  o  o  o  o  o  o  o  o
o  o  o  o  o  o  o  o  o  o  o  o
o  o  o  o  o  o  o  o  o  o  o  o
x  C  C  C  C  C  C  C  C  C  C  T
```

Options

We custom define very simple options here. They might not be the logical options for this settings deliberately chosen to visualise the Q Table better.

```
# We are defining two more options here
# Option 1 ["Away"] - > Away from Cliff (ie keep going up)
# Option 2 ["Close"] - > Close to Cliff (ie keep going down)

def Away(env,state):

    optdone = False
    optact = 0

    if (int(state/12) == 0):
        optdone = True

    return [optact,optdone]

def Close(env,state):

    optdone = False
    optact = 2

    if (int(state/12) == 2):
        optdone = True

    return [optact,optdone]


'''
Now the new action space will contain
Primitive Actions: ["up", "right", "down", "left"]
Options: ["Away","Close"]
Total Actions :["up", "right", "down", "left", "Away", "Close"]
Corresponding to [0,1,2,3,4,5]
'''
```

```
{"type":"string"}
```

Task 1

Complete the code cell below

```
# epsilon-greedy action selection function
def egreedy_policy(q_values, state, epsilon):
    if np.random.rand() < epsilon:
        return np.random.randint(6)
    else:
        return np.argmax(q_values[state])
```

Task 2

Below is an incomplete code cell with the flow of SMDP Q-Learning. Complete the cell and train the agent using SMDP Q-Learning algorithm. Keep the **final Q-table** and **Update Frequency** table handy (You'll need it in TODO 4)

```
#### SMDP Q-Learning
update_frequency_smdp = np.zeros((env.nS, 6))  # For tracking update
frequency
cumulative_rewards_smdp = []  # For tracking cumulative rewards
# Parameters
gamma = 0.9
alpha = 0.1
epsilon = 0.1

# Q-Values initialization
q_values_SMDP = np.zeros((48, 6))

# SMDP Q-Learning with Update Frequency
for episode in range(1000):
    state = env.reset()
    done = False
    total_reward = 0

    for _ in range(1000):
        action = egreedy_policy(q_values_SMDP, state, epsilon)

        if action < 4:  # Primitive action
            next_state, reward, done, _ = env.step(action)
            best_next_action = np.argmax(q_values_SMDP[next_state])
            q_values_SMDP[state, action] += alpha * (reward + gamma *
q_values_SMDP[next_state, best_next_action] - q_values_SMDP[state,
```

```
action])
            update_frequency_smdp[state, action] += 1  # Update
frequency
            state = next_state
        else:
            reward_bar = 0
            beta = 1  # Discounting over steps within the option
            optdone = False
            # while not optdone and not done:
            for _ in range(1000):
                if action == 4:  # "Away" option
                    optact, optdone = Away(env, state)
                elif action == 5:  # "Close" option
                    optact, optdone = Close(env, state)

                next_state, reward, done, _ = env.step(optact)
                reward_bar += beta * reward
                beta *= gamma
                state = next_state
                if done or optdone:
                  break

            q_values_SMDP[state, action] += alpha * (reward_bar -
q_values_SMDP[state, action])
            update_frequency_smdp[state, action] += 1  # Update
frequency

        total_reward += reward
        if done:
          break

    cumulative_rewards_smdp.append(total_reward)
```

Task 3

Using the same options and the SMDP code, implement Intra Option Q-Learning (In the code cell below). You *might not* always have to search through options to find the options with similar policies, think about it. Keep the **final Q-table** and **Update Frequency** table handy (You'll need it in TODO 4)

```
#### Intra-Option Q-Learning
q_values_IOQL = np.zeros((48, 6))
update_frequency_ioql = np.zeros((env.nS, 6))  # For tracking update
frequency
cumulative_rewards_intra_option = []  # For tracking cumulative
rewards
```

```
# Intra-Option Q-Learning with Update Frequency
for episode in range(1000):
    state = env.reset()
    done = False
    total_reward = 0

    for i in range(1000):
        action = egreedy_policy(q_values_IOQL, state, epsilon)

        if action < 4:  # Primitive action
            next_state, reward, done, _ = env.step(action)
            best_next_action = np.argmax(q_values_IOQL[next_state])
            q_values_IOQL[state, action] += alpha * (reward + gamma *
q_values_IOQL[next_state, best_next_action] - q_values_IOQL[state,
action])
            update_frequency_ioql[state, action] += 1  # Update
frequency
            state = next_state
        else:
            optdone = False
            # while not optdone and not done:
            for _ in range(1000):
                if action == 4:  # "Away" option
                    optact, optdone = Away(env, state)
                elif action == 5:  # "Close" option
                    optact, optdone = Close(env, state)

                next_state, reward, done, _ = env.step(optact)
                best_next_action =
np.argmax(q_values_IOQL[next_state])
                q_values_IOQL[state, action] += alpha * (reward +
gamma * q_values_IOQL[next_state, best_next_action] -
q_values_IOQL[state, action])
                update_frequency_ioql[state, action] += 1  # Update
frequency
                state = next_state
                if done or optdone:
                  break

        total_reward += reward
        if done:
          break

    cumulative_rewards_intra_option.append(total_reward)
```

Task 4

Compare the two Q-Tables and Update Frequencies and provide comments.

```python
# Use this cell for Task 4 Code# Task 4
# Compare the two Q-Tables and Update Frequencies and provide
comments.

# Compare final Q-tables
print("Comparison of final Q-tables:")
print("SMDP Q-Learning Q-table:")
print(q_values_SMDP)
print("\nIntra-Option Q-Learning Q-table:")
print(q_values_IOQL)

# Compare update frequencies
print("\nComparison of update frequencies:")
print("SMDP Q-Learning Update Frequency:")
print(update_frequency_smdp)
print("\nIntra-Option Q-Learning Update Frequency:")
print(update_frequency_ioql )
```

```
Comparison of final Q-tables:
SMDP Q-Learning Q-table:
[[  -1.           -1.           -1.           -1.           -
3.42462733
     0.        ]
 [  -0.99999983   -0.99999999   -0.99999994   -0.99999952   -
1.98603141
     0.        ]
 [  -0.271        -0.19         -0.3439       -0.5217031    -
1.19542509
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1          -0.1
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.1          -0.1          -0.1          -0.1           0.
     0.        ]
 [  -0.99999946   -0.99999738   -0.99999961   -0.99999985    0.
     0.        ]
 [  -0.40951      -0.5217031    -0.61257951   -0.1           0.
```

```
     0.         ]
[  -0.1          -0.1          -0.19         -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.1          -0.1          -0.1          -0.1           0.
     0.         ]
[  -0.99999709   -0.99999986   -0.99999993   -0.9999994     0.
     0.         ]
[  -0.271        -0.271       -40.951        -0.19          0.
     0.         ]
[  -0.19         -0.1         -10.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1           0.            0.            0.
     0.         ]
[  -0.1          -0.1          -0.1           0.            0.
     0.         ]
[  -1.         -100.           -1.           -1.            0.
    -3.29339625]
[   0.            0.            0.            0.            0.
     0.         ]
```

```
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]
 [   0.              0.              0.              0.              0.
     0.          ]]

Intra-Option Q-Learning Q-table:
[[  -7.66813894   -7.63634765   -7.63625949   -7.71313663   -7.869422
    -7.63833348]
 [  -7.41242549   -7.37927809   -7.37922929   -7.53451083   -
7.53713038
    -7.37945131]
 [  -7.14543252   -7.09508646   -7.09832242   -7.15982086   -
7.30063433
    -7.09610397]
 [  -6.81151791   -6.77965826   -6.78562903   -7.01178753   -
6.95936416
    -6.78255027]
 [  -6.51634863   -6.43005813   -6.44217148   -6.58814186   -
6.53726007
    -6.43330216]
 [  -6.05083316   -6.04690732   -6.05034874   -6.25751876   -
6.34267214
    -6.04873472]
 [  -5.71605594   -5.62142424   -5.62043878   -5.71438801   -
5.97383286
    -5.62504892]
 [  -5.17219261   -5.1483253    -5.1485301    -5.15868614   -
5.31928706
    -5.1528261 ]
 [  -4.67164374   -4.62691402   -4.62890536   -4.68969221   -
4.97990226
    -4.62652174]
 [  -4.05258316   -4.04658699   -4.05041125   -4.32209072   -
```

```
4.47944405
    -4.04883142]
 [  -3.42930797   -3.40232993   -3.40351087   -3.47865585   -
3.70537054
    -3.40549993]
 [  -2.74206096   -2.74724097   -2.69344517   -2.80434205   -
3.13190614
    -2.69430473]
 [  -7.45403854   -7.40812062   -7.41026511   -7.44186478   -
7.86941428
    -7.45810456]
 [  -7.22897491   -7.14371613   -7.14410723   -7.15209677   -
7.49119702
    -7.17453148]
 [  -6.86026055   -6.8377241    -6.83990758   -6.93867095   -
7.26684627
    -6.85794495]
 [  -6.63311487   -6.49477012   -6.49571216   -6.75768113   -
6.85954314
    -6.50818899]
 [  -6.15960045   -6.11247306   -6.11327803   -6.14433361   -6.4031369
    -6.11721195]
 [  -5.78325174   -5.68568769   -5.68570948   -5.83113348   -
6.24751991
    -5.68810706]
 [  -5.37860746   -5.20970643   -5.20944492   -5.40822225   -5.9267941
    -5.21007215]
 [  -4.88277009   -4.68030096   -4.68023199   -4.87804781   -
5.16439596
    -4.6804695 ]
 [  -4.46598678   -4.09149513   -4.09167871   -4.25931402   -
4.85374364
    -4.09187752]
 [  -3.51985291   -3.43701087   -3.43714254   -3.83105002   -
4.37315371
    -3.43722044]
 [  -2.74082055   -2.7092188    -2.70921781   -2.84579691   -
3.44729981
    -2.70935282]
 [  -2.05364586   -1.98342513   -1.89996389   -2.00589225   -
2.96484354
    -1.89996401]
 [  -7.57374759   -7.17570464   -7.61726774   -7.41202251   -
7.66142626
    -7.71231526]
 [  -7.28762082   -6.86189404 -100.27068387   -7.27309483   -
7.27256092
  -106.71208967]
 [  -7.00560694   -6.5132156   -98.91721169   -7.08271437   -
```

```
7.02644226
  -106.71206116]
 [  -6.6664285    -6.12579511 -101.49643683   -6.76441556   -
6.64555367
  -106.71125183]
 [  -6.23055555   -5.6953279  -101.97295985   -6.41142653   -
6.16655613
  -106.71062512]
 [  -5.96235878   -5.217031    -79.29823843   -6.00678725   -
5.91753485
  -106.70783254]
 [  -5.54250895   -4.68559     -96.08627813   -5.47242486   -
5.58298844
  -106.70881785]
 [  -5.05468215   -4.0951      -95.95800879   -4.97367844   -
4.82563869
  -106.70962356]
 [  -4.38531276   -3.439       -98.14505633   -4.34800172   -
4.40236926
  -106.70926872]
 [  -3.81209791   -2.71        -93.63845325   -3.96715699   -
3.86884252
  -106.71106593]
 [  -3.17487719   -1.9         -82.22014735   -3.04180755   -
2.85442322
  -106.71091825]
 [  -2.5360875    -1.73996286   -1.           -2.35659088   -
2.29632296
    -1.        ]
 [  -7.45813417 -106.59717053   -7.69579861   -7.70563347   -
7.45813417
    -7.71232075]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
    0.        ]
 [   0.            0.            0.            0.            0.
```

```
     0.        ]
 [   0.            0.            0.            0.            0.
     0.        ]
 [   0.            0.            0.            0.            0.
     0.        ]]

Comparison of update frequencies:
SMDP Q-Learning Update Frequency:
[[8.21200e+03 8.21700e+03 8.00500e+03 8.15700e+03 4.69911e+05
0.00000e+00]
 [1.48000e+02 1.73000e+02 1.58000e+02 1.38000e+02 5.40000e+02
0.00000e+00]
 [3.00000e+00 2.00000e+00 4.00000e+00 7.00000e+00 1.20000e+01
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.37000e+02 1.22000e+02 1.40000e+02 1.49000e+02 0.00000e+00
0.00000e+00]
 [5.00000e+00 7.00000e+00 9.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 2.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
```

```
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00
0.00000e+00]
 [1.21000e+02 1.50000e+02 1.56000e+02 1.36000e+02 0.00000e+00
0.00000e+00]
 [3.00000e+00 3.00000e+00 5.00000e+00 2.00000e+00 0.00000e+00
0.00000e+00]
 [2.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [1.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [8.13800e+03 8.08200e+03 8.08600e+03 8.07400e+03 0.00000e+00
4.62054e+05]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
```

```
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]
 [0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
0.00000e+00]]

Intra-Option Q-Learning Update Frequency:
[[1.470e+02 1.050e+03 3.180e+02 1.490e+02 1.528e+03 3.160e+02]
 [1.360e+02 8.760e+02 2.110e+02 8.600e+01 1.760e+02 2.100e+02]
 [1.260e+02 7.620e+02 1.740e+02 7.600e+01 1.610e+02 1.740e+02]
 [1.150e+02 6.520e+02 1.520e+02 7.500e+01 1.370e+02 1.510e+02]
 [1.060e+02 5.570e+02 1.370e+02 6.600e+01 1.220e+02 1.350e+02]
 [9.300e+01 4.720e+02 1.220e+02 6.100e+01 1.140e+02 1.210e+02]
 [8.500e+01 4.020e+02 1.090e+02 5.300e+01 1.090e+02 1.090e+02]
 [7.300e+01 3.240e+02 9.800e+01 4.600e+01 8.100e+01 9.800e+01]
 [6.300e+01 2.580e+02 8.800e+01 4.100e+01 7.800e+01 8.700e+01]
 [5.200e+01 1.840e+02 7.900e+01 3.700e+01 7.000e+01 7.800e+01]
 [4.200e+01 1.050e+02 7.000e+01 2.900e+01 5.100e+01 7.000e+01]
 [3.200e+01 3.200e+01 6.500e+01 2.500e+01 4.200e+01 6.500e+01]
 [9.600e+01 3.640e+02 1.170e+02 1.360e+02 1.502e+03 3.290e+02]
 [1.000e+02 4.380e+02 1.320e+02 8.200e+01 1.580e+02 2.250e+02]
 [9.500e+01 4.550e+02 1.390e+02 7.400e+01 1.400e+02 1.870e+02]
 [9.200e+01 4.410e+02 1.390e+02 7.000e+01 1.100e+02 1.620e+02]
 [8.200e+01 4.070e+02 1.350e+02 5.900e+01 1.000e+02 1.510e+02]
 [7.400e+01 3.700e+02 1.280e+02 5.500e+01 9.300e+01 1.380e+02]
 [6.700e+01 3.290e+02 1.200e+02 4.900e+01 8.800e+01 1.270e+02]
 [5.800e+01 2.840e+02 1.120e+02 4.200e+01 6.600e+01 1.170e+02]
 [5.100e+01 2.300e+02 1.040e+02 3.600e+01 6.200e+01 1.080e+02]
 [3.700e+01 1.730e+02 9.800e+01 3.200e+01 5.400e+01 1.020e+02]
 [2.700e+01 1.110e+02 9.400e+01 2.300e+01 3.800e+01 9.900e+01]
 [1.900e+01 2.200e+01 1.130e+02 1.700e+01 3.100e+01 1.140e+02]
 [1.980e+02 1.702e+03 9.000e+01 1.440e+02 1.494e+03 3.670e+02]
 [1.380e+02 1.505e+03 2.700e+01 7.900e+01 1.380e+02 2.530e+02]
 [1.160e+02 1.337e+03 2.500e+01 8.100e+01 1.180e+02 2.300e+02]
 [9.900e+01 1.211e+03 2.900e+01 7.600e+01 9.800e+01 1.960e+02]
 [8.400e+01 1.116e+03 3.000e+01 7.100e+01 8.200e+01 1.790e+02]
 [8.100e+01 1.047e+03 1.300e+01 6.500e+01 7.800e+01 1.520e+02]
 [7.400e+01 9.730e+02 2.200e+01 5.400e+01 7.600e+01 1.430e+02]
 [6.400e+01 9.200e+02 2.200e+01 4.800e+01 5.400e+01 1.340e+02]
 [5.000e+01 8.920e+02 2.400e+01 4.000e+01 5.000e+01 1.200e+02]
 [4.200e+01 8.480e+02 2.000e+01 4.400e+01 4.400e+01 1.220e+02]
 [3.500e+01 8.460e+02 1.400e+01 2.800e+01 2.700e+01 1.140e+02]
 [3.100e+01 2.700e+01 6.370e+02 2.300e+01 2.200e+01 3.600e+02]
 [1.861e+03 6.800e+01 1.650e+02 1.730e+02 1.465e+03 6.800e+04]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
```

```
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00]]

Comments:
1. Convergence Speed: Check how quickly each algorithm's Q-table
converges towards optimal values. Faster convergence indicates better
learning.
2. Learned Policies: Analyze the learned policies from both
algorithms. Are they similar or different?
3. Exploration of State-Action Space: Determine how efficiently each
algorithm explores the state-action space. Do they cover all possible
actions and states?
4. Utilization of Options: Assess how effectively each algorithm
utilizes the defined options in the environment. Do options improve
learning performance?
```

Use this text cell for your comments - Task 4

Inference

Final Q-Tables Analysis:

- **SMDP Q-Learning Q-table** suggests a strong distinction between the values assigned to primitive actions (up, right, down, left) and the two options ("Away" and "Close"). For many states, the options have significantly different Q-values compared to primitive actions, indicating that the algorithm has learned when it's advantageous to execute these options.

- **Intra-Option Q-Learning Q-table** shows a more varied distribution of Q-values across both primitive actions and options. This indicates a more dynamic use of both options and primitive actions across different states.

Update Frequencies Analysis:

- **SMDP Q-Learning Update Frequency** shows a high frequency of updates for the options compared to primitive actions in certain states, which indicates that the options were utilized extensively during the learning process.

- **Intra-Option Q-Learning Update Frequency** displays a more balanced update frequency between primitive actions and options, which indicates more exploratory behaviore not just with options but also with primitive actions.

Inference:

- **SMDP Q-Learning** seems to prioritize options, potentially at the expense of exploring primitive actions, which might limit policy flexibility.
- **Intra-Option Q-Learning** provides a more balanced and nuanced approach, leveraging both options and primitive actions for a potentially more adaptable and refined policy.
- The significant difference in update frequencies and Q-values across the two methods highlights the trade-off between focusing on high-level strategies (options) versus detailed, action-level decisions in hierarchical reinforcement learning.